CLAIMS

1. A user interface for inserting a custom algorithm in a data-mining application, the user interface comprising:

a control to upload algorithm code;

a control to query the user for input and output parameter information;

wherein the user interface is available to pass the algorithm source code to an evaluation process, the evaluation process being available to determine whether the user has properly implemented interface requirements; and

wherein the user interface is available to pass the algorithm source code to a wrapping process that wraps the algorithm in an appropriate language-specific accessor function.

2. The user interface according to claim 1 wherein the algorithm source code is written in a high level-language.

3. The user interface according to claim 2 wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

4. The user interface according to claim 1 wherein the control to upload an algorithm source code is a single control element.

5. The user interface according to claim 1 wherein the control to upload an algorithm source code is a plurality of elements comprising a text box in which to identify a file, a browse button with which to select a file, and an upload button with which to initiate the upload process.

6. The user interface according to claim 1 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

7. The user interface according to claim 1 wherein the interface requirements checked by the evaluation process include an entry point into the code and exit state.

8. The user interface according to claim 1 wherein the wrapping process is a back-end procedure.

9. A method for inserting a custom algorithm in a data-mining application, the method comprising:

uploading an algorithm source code;

receiving input and output parameter information from the user;

evaluating the algorithm source code to determine whether the user has properly implemented interface requirements; and

passing the algorithm source code to a wrapping process that wraps the algorithm in an appropriate language-specific accessor function.

10. The method for inserting a custom algorithm in a data-mining application according to claim 9 wherein the algorithm source code is written in a high level-language.

11. The method for inserting a custom algorithm in a data-mining application according to claim 10 wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

12. The method for inserting a custom algorithm in a data-mining application according to claim 9 wherein the processes are tied to a user interface.

13. The method for inserting a custom algorithm in a data-mining application according to claim 9 wherein processes are performed by a separate application.

14. The method for inserting a custom algorithm in a data-mining application according to claim 9 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

15. The method for inserting a custom algorithm in a data-mining application according to claim 9 wherein the interface requirements evaluated include an entry point into the code and exit state.

16. The method for inserting a custom algorithm in a data-mining application according to claim 9 wherein the wrapping process is a back-end procedure.

17. An interface for inserting a customer algorithm into a data-mining application, the interface comprising:

a means for uploading an algorithm source code;

a means for receiving input and output parameter information from the user;

a means for evaluating the algorithm source code to determine whether the user has properly implemented interface requirements; and

a means for passing the algorithm source code to a wrapping process that wraps the algorithm in an appropriate language-specific accessor function.

18. The interface for inserting a custom algorithm in a data-mining application according to claim 17 wherein the algorithm source code is written in a high level-language.

19. The interface for inserting a custom algorithm in a data-mining application according to claim 18 wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

20. The interface for inserting a custom algorithm in a data-mining application according to claim 17 wherein the means are contained in a user interface.

21. The interface for inserting a custom algorithm in a data-mining application according to claim 17 wherein means are contained in a separate application.

22. The interface for inserting a custom algorithm in a data-mining application according to claim 17 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

23. The interface for inserting a custom algorithm in a data-mining application according to claim 17 wherein the interface requirements evaluated include an entry point into the code and exit state.

24. The interface for inserting a custom algorithm in a data-mining application according to claim 17 wherein the wrapping process is a back-end procedure.

25. An article of manufacture for inserting a customer algorithm into an analysis environment, comprising a computer readable media containing:

a computer program code segment that uploads an algorithm source code;

a computer program code segment that receives input and output parameter information from the user;

a computer program code segment that evaluates the algorithm source code to determine whether the user has properly implemented interface requirements; and

a computer program code segment that passes the algorithm source code to a wrapping process that wraps the algorithm in an appropriate language-specific accessor function.

26. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 25 wherein the algorithm source code is written in a high level-language.

27. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 26 wherein the high-level language is selected from the group

consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

28. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 25 wherein the computer readable medium further comprises a user interface comprising the computer program code segments.

29. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 25 wherein computer program code segments are part of a separate application.

30. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 25 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

31. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 25 wherein the interface requirements evaluated include an entry point into the code and exit state.

32. The article of manufacture for inserting a custom algorithm in a data-mining application according to claim 25 wherein the wrapping process is a back-end procedure.

33. A data-mining computer system adapted for inserting a custom algorithm into the data mining application, comprising:

an upload control that uploads an algorithm source code;

a parameter control that receives input and output parameter information from the user;

an evaluation process that evaluates the algorithm source code to determine whether the user has properly implemented interface requirements; and

a wrapping process that wraps the algorithm in an appropriate language-specific accessor function.

34. The data-mining computer system according to claim 33 wherein the algorithm source code is written in a high level-language.

35. The data-mining computer system according to claim 34 wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

36. The data-mining computer system according to claim 33 further comprising a user interface comprising the upload control and the parameter control.

37. The data-mining computer system according to claim 33 wherein the upload control and the parameter control are inputs for an application.

38. The data-mining computer system according to claim 33 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

39. The data-mining computer system according to claim 33 wherein the evaluation process evaluates an entry point into the code and exit state.

40. The data-mining computer system according to claim 33 wherein the wrapping process is a back-end procedure.

41. A client system adapted for inserting a custom algorithm into a data-mining application, the client system comprising:

an upload control that uploads an algorithm source code;

a parameter control that receives input and output parameter information from the user;

an evaluation process link that can call an evaluation process available to evaluate the algorithm source code to determine whether the user has properly implemented interface requirements; and

a wrapping process link that can call a wrapping process available to wrap the algorithm in an appropriate language-specific accessor function.

42. The client system according to claim 41 wherein the algorithm source code is written in a high level-language.

43. The client system according to claim 42 wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

44. The client system according to claim 41 further comprising a user interface comprising the upload control and the parameter control.

45. The client system according to claim 41 wherein the upload control and the parameter control each present a prompt to the user and receive user input.

46. The client system according to claim 41 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

47. The client system according to claim 41 wherein the evaluation process evaluates an entry point into the code and exit state.

48. The client system according to claim 41 wherein the wrapping process is a back-end procedure.

49. A server system wherein a custom algorithm can be inserted into an analysis environment, the server system comprising:

an upload control that uploads an algorithm source code;

a parameter control that receives input and output parameter information from the user;

an evaluation process link that can call an evaluation process available to evaluate the algorithm source code to

determine whether the user has properly implemented interface requirements; and

a wrapping process link that can call a wrapping process available to wrap the algorithm in an appropriate language-specific accessor function.

50. The server system according to claim 49 wherein the algorithm source code is written in a high level-language.

51. The server system according to claim 50 wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

52. The server system according to claim 49 further comprising a user interface comprising the upload control and the parameter control.

53. The server system according to claim 49 wherein the upload control and the parameter control each present a prompt to the user and receive user input.

54. The server system according to claim 49 wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

55. The server system according to claim 49 wherein the evaluation process evaluates an entry point into the code and exit state.

56. The server system according to claim 49 wherein the wrapping process is a back-end procedure.

57. A computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application, the computer program comprising instructions for performing the method of claim 9.

58. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm

in a data-mining application according to claim 57, wherein the algorithm source code is written in a high level-language.

59. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application according to claim 58, wherein the high-level language is selected from the group consisting of C, C++, Java, Matlab, Fortran, Pascal, and Visual Basic.

60. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application according to claim 57, wherein the processes are tied to a user interface.

61. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application according to claim 57, wherein processes are performed by a separate application.

62. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application according to claim 57, wherein the input and output parameter information comprises data format, default values, help dialogs, and parameter relationships.

63. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application according to claim 57, wherein the interface requirements evaluated include an entry point into the code and exit state.

64. The computer data signal embodied in a carrier wave encoding a computer program for inserting a custom algorithm in a data-mining application according to claim 57, wherein the wrapping process is a back-end procedure.

65. A method of providing a ground truth tool in a database having data fields, comprising:

processing to detect, to cluster, and to track contiguous events;

presenting detected, clustered, and tracked contiguous events in groups wherein the members of each group have similar characteristics; and

receiving input assigning class labels to the events.

66. The method of providing a ground truth tool according to claim 65 wherein the processing is digital signal processing to detect, to cluster, and to track temporally contiguous events.

67. The method of providing a ground truth tool according to claim 65 wherein the processing is image processing to detect, to cluster, and to track spatially contiguous events.

68. The method of providing a ground truth tool according to claim 65 further comprising storing the class labels in a new data field appended the database.

69. The method of providing a ground truth tool according to claim 65 wherein events are presented and input is received on controls of a user interface.

70. A computer program storage medium readable by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 65.

71. A computer program storage medium readable by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 66.

72. A computer program storage medium readable by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 67.

73. A computer program storage medium readable by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 68.

74. A computer program storage medium readable by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 69.

75. A computer data signal embodied in a carrier wave by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions to perform the method of claim 65.

76. A computer data signal embodied in a carrier wave by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 66.

77. A computer data signal embodied in a carrier wave by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 67.

78. A computer data signal embodied in a carrier wave by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 68.

79. A computer data signal embodied in a carrier wave by a computing system and encoding a computer program for providing a ground truth tool, the computer program comprising instructions for performing the method of claim 69.

80. A computer system having a data-mining application and including a ground truth tool, the system comprising:

means for detecting, clustering, and tracking contiguous events;

means for presenting detected, clustered, and tracked contiguous events in groups wherein the members of each group have similar characteristics;

means for receiving input assigning class labels to the events.

81. The computer system according to claim 80 wherein the means for detecting, clustering, and tracking contiguous events is a digital signal processor to detect, to cluster, and to track temporally contiguous events.

82. The computer system according to claim 80 wherein the means for detecting, clustering, and tracking contiguous events is an image processor to detect, to cluster, and to track spatially contiguous events.

83. The computer system according to claim 80 further comprising a means for storing the class labels in a new data field appended the database.

84. The computer system according to claim 80 wherein events are presented and input is received on controls of a user interface.

85. A method for seamless insertion of custom algorithms in a data-mining application using tap points, the method comprising:

using a computer system for machine-assisted problem exploration in a data-mining application, the computer system having a problem-definition user interface;

concluding that additional operations are needed that are too complicated to be specified easily using the problem-definition interface;

displaying to the user all data-mining steps and a tap-point dissemination helper; and

receiving input from the user specifying when to extract an intermediate output for further processing.

86. The method according to claim 85 wherein the tap points are file-based.

87. The method according to claim 85 wherein the tap points are not file-based.

88. The method according to claim 85 wherein the machines-assisted problem definition uses a Bayesian network.

89. The method according to claim 85 wherein the machines-assisted problem definition uses a decision tree.

90. The method according to claim 85 wherein the displaying step and the receiving input step use a user interface.

91. The method according to claim 85 wherein user input specifies the format in which data will output.

92. A user interface adapted for specifying data tap-points in a data-mining application, the interface comprising:

an output that displays information about the data-mining steps and a tap-point dissemination helper; and

an input that receives information from the user to specify when to extract an intermediate output for further processing.

93. The user interface according to claim 92 wherein the output is a control on a user interface and the input is a control on a user interface.

94. The user interface according to claim 92 wherein intermediate output is extracted at file-based tap points identified by the user.

95. A computer readable medium comprising instructions for seamless insertion of custom algorithms in a data-mining application using tap points, said instructions comprising the acts of:

using a computer system for machine-assisted problem exploration in a data-mining application, the computer system having a problem-definition user interface;

concluding that additional operations are needed that are too complicated to be specified easily using the problem-definition interface;

displaying to the user all data-mining steps and a tap-point dissemination helper; and

receiving input from the user specifying when to extract an intermediate output for further processing.

96. The computer readable medium according to claim 95 wherein the tap points are file-based.

97. The computer readable medium according to claim 95 wherein the tap points are not file-based.

98. The computer readable medium according to claim 95 wherein the machines-assisted problem definition uses a Bayesian network.

99. The computer readable medium according to claim 95 wherein the machines-assisted problem definition uses a decision tree.

100. The computer readable medium according to claim 95 wherein the displaying step and the receiving input step use a user interface.

101. The computer readable medium according to claim 95 wherein user input specifies the format in which data will output.

102. A computer data signal embodied in a carrier wave and representing sequences of instructions which, when executed by a processor, cause said processor to seamlessly insert a custom algorithms in a data-mining application using tap points by performing the steps of:

using a computer system for machine-assisted problem exploration in a data-mining application, the computer system having a problem-definition user interface;

concluding that additional operations are needed that are too complicated to be specified easily using the problem-definition interface;

displaying to the user all data-mining steps and a tap-point dissemination helper; and

receiving input from the user specifying when to extract an intermediate output for further processing.

103. The computer data signal according to claim 102 wherein the tap points are file-based.

104. The computer data signal according to claim 102 wherein the tap points are not file-based.

105. The computer data signal according to claim 102 wherein the machines-assisted problem definition uses a Bayesian network.

106. The computer data signal according to claim 102 wherein the machines-assisted problem definition uses a decision tree.

107. The computer data signal according to claim 102 wherein the displaying step and the receiving input step use a user interface.

108. The computer data signal according to claim 102 wherein user input specifies the format in which data will output.

109. A computer system including means for seamless insertion of custom algorithms in a data-mining application using tap points, the computer system comprising:

means for using a computer system for machine-assisted problem exploration in a data-mining application, the computer system having a problem-definition user interface;

means for concluding that additional operations are needed that are too complicated to be specified easily using the problem-definition interface;

means for displaying to the user all data-mining steps and a tap-point dissemination helper; and

means for receiving input from the user specifying when to extract an intermediate output for further processing.

110. The computer system according to claim 109 wherein the tap points are file-based.

111. The computer system according to claim 109 wherein the tap points are not file-based.

112. The computer system according to claim 109 wherein the machines-assisted problem definition uses a Bayesian network.

113. The computer system according to claim 109 wherein the machines-assisted problem definition uses a decision tree.

114. The computer system according to claim 109 wherein the displaying means and the receiving input means comprise a user interface.

115. The computer system according to claim 109 wherein user input specifies the format in which data will output.

116. A computer system including seamless insertion of custom algorithms in a data-mining application using tap points, the computer system comprising:

a memory and a central processor;

a machine-assisted problem exploration processor in a data-mining application;

an output device, the output device communicating data-mining steps and a tap-point dissemination helper; when additional operations are needed that are too complicated to be specified easily using the machine-assisted problem exploration processor; and

an input device for receiving input from the user specifying when to extract an intermediate output for further processing.

117. The computer system according to claim 116 wherein the output device is a member of the group consisting of a cathode ray tube and a printer.

118. The computer system according to claim 116 wherein the input device is a keyboard.